Abstract of 25th Annual Software Engineering Workshop Paper: "Enhanced Mission Success with Software Development Principles: Application to Mars'98 Missions"

Milton Lavin, Jet Propulsion Laboratory, Pasadena, CA 91109 8/21/00

Background

The objective of this paper is to illustrate the use of experience-based development principles in order to identify and eliminate software defects prior to launch. The application context is selected from software problems contributing to two recent NASA mission losses -- the Mars Climate Orbiter (MCO) Mission in October 1999 and the Mars Polar Lander (MPL) Mission in December 1999. In the case, of MCO, the failure to achieve Martian orbit was traced to a units error in the small forces software used in navigation. The presumed cause of the MPL loss during descent is a defect in the software controlling the touchdown sensor. In each case, the application is presented with the intent of demonstrating the synergy and effectiveness of employing a comprehensive set of software development practices to root out defects. Because the MCO and MPL Loss Reports were inputs to the software development principles, these two applications cannot be construed as a validation of the methodology.

Effective software development is dependent both on a mature development process (standards, procedures and policies) and on the continual incorporation into the process of lessons learned from historical projects. These "lessons learned" are typically more detailed than general principles of software engineering because they are idiosyncratic to each development context -- in JPL's case, robotic missions into deep space. It is particularly important to document experience and incorporate it into the development process when many project managers have limited experience in software development. Tighter development schedules and reduced budgets are another factor that stimulates the search for more effective ways to build quality into the product while satisfying programmatic demands. Thus for several reasons, it was decided to make these lessons learned more accessible to the JPL development community by 1) analyzing key factors in recent mission successes and failures and 2) distilling the findings into concise principles to be observed in planning and implementing a software development.

Development of Software Development Principles for Flight Systems

In conjunction with a broader initiative to document good system development practice, The Center for Space Mission Information and Software Systems at the Jet Propulsion Laboratory (JPL) has developed a set of 103 software development principles for flight systems, organized around life cycle activities. Both in-house development and software acquisition are addressed. A development principle is understood to be a best practice to which exception can be taken for good reason -- but only for good reason. Adherence is verified by 1) documenting deviations in the software management plan and 2) using these principles to probe development process details during reviews of plans, requirements, designs, and test results. These principles are based on experience in recent flight projects -- both successes and failures. Primary sources were discussions with software managers, system engineers, and developers spanning all major software-intensive mission systems, plus the findings and lessons learned in the six cited sources

-- three investigations of the two recent Mars Mission losses, two reports that draw more broadly on flight software development at JPL and GSFC, and NASA's Lessons Learned website.

The process of developing these software principles was iterative, with each version subject to extensive peer review by developers, subject matter experts, line managers, and project managers. In the early phase of development, the selection of principles was guided by an editorial board, armed with evaluation criteria. After the first workshop review, the principles were baselined, and further changes were approved by a change board. It is expected that projects in the planning and early implementation phases will be the initial users of these software principles -- Mars Rover'03, and Outer Planets/Solar Probe are two examples.

Application of Software Development Principles to Mars'98 Missions

In the case of both the MCO units error and the software logic controlling the MPL touchdown sensor, the defect was introduced in the flowdown of systems requirements to software requirements. Exhibit 1 summarizes the multiple opportunities to identify and correct each defect via the application of cited software principles -- beginning with planning for peer reviews and ending with pre-delivery and acceptance testing. An "X" in Exhibit 1 denotes that the indicated activity was either missing or ineffectively implemented.

Exhibit 1: Application of Software Development Principles to Mars Climate Orbiter and Mars Polar Lander Software Defects

Software Development Principle	**MCO**	**MPL**
3.2 Planning and Monitoring		
3.2.7 Joint development planning for interfacing HW & SW		X
3.2.9 Identification of milestone and peer reviews	X	X
3.2.10 Participation of HW engineers and operations in reviews		X
3.2.12 Peer review of intermediate products	X	X
3.4 Risk Management		
3.4.3 Early validation of interfaces, high-risk algorithms, COTS	X	X
3.6 Design and Implementation		
3.6.3 Design traced to software & mission requirements	X	X
3.6.4 Analytical basis for logic design		X
3.6.6 SW logic to verify values of input & output parameters	X	X
3.7 Integration and Test		
3.7.6 Detailed testing of mission phase transitions		X
3.7.7 Testing to address FTA and off-nominal HW behavior		X
3.7.9 Aggressively find latent defects via stress testing	X	X
3.7.13 Trace from final system test to mission requirements	X	X
3.9 Software Acquisition		
3.9.1 PIP to address management of software acquisition: • In-process JPL review of intermediate products • JPL participation in pre-delivery testing	 X	 X X
3.10 Product and Process Verification		
3.10.4 Acceptance test exercising mission-critical systems	X	X
4.0 Flight Software		
4.5 Accommodation of nominal, off-nominal/transient inputs		X

Because software creation is complex and subject to a variety of human error in specification, design, and implementation, it is not surprising that serious defects were introduced into the MCO and MPL software during development. The activities suitable for defect identification and removal are well known, but they must be applied systematically as an ensemble because in practice each is only partially effective.

References:

J. Casani et al., Report on the Loss of the Mars Climate Orbiter Mission, JPL D-18441 (November 1999)

J. Casani, et al., Report on the Loss of the Mars Polar Lander and Deep Space 2 Missions, JPL D-18709 (March 2000)

J. Hihn and H. Habib-agahi, Flight Software Cost Growth: Causes and Recommendations, JPL D-18660 (February 2000)

C. Lin and H. Kea, GSFC/JPL Quality Mission Software Workshop Report (October 1999)

NASA Lessons Learned Information System site -- http://llis.nasa.gov/

J. Vellinga, Mars Polar Lander (MPL) Possible Premature Engine Thrust Termination Process Investigation Report, Lockheed Martin Space Systems, Denver Operations, MSP-00-5001 (March 2000)